Submission in Response to NSF CI 2030 Request for Information

DATE AND TIME: 2017-03-08 11:01:46
REFERENCE NO: 180

Author Names & Affiliations

- Thomas Cusick - University at Buffalo, Mathematics Dept., Buffalo, NY 14260

Contact Email Address (for NSF use only)

(Hidden)

Research Domain, discipline, and sub-discipline

Mathematics, cryptography

Title of Submission

New software and hardware for cybersecurity

Abstract (maximum ~200 words).

People are the weak link in cybersecurity. Vetting is not enough to prevent malicious employees from obtaining security clearances. Even if an employee is honest, that person may be deceived by a phishing attack or a cleverly designed domain name homograph attack. New ideas in software ("permission software," as described in the submission) and hardware (a special type of port, like the standard USB port, for regulating access to government computers) are needed.

Question 1 Research Challenge(s) (maximum ~1200 words): Describe current or emerging science or engineering research challenge(s), providing context in terms of recent research activities and standing questions in the field.

Question 1: It is very urgent that a method for defense against phishing attacks on government computers be implemented. It is not enough to advise users what to do. Users are sometimes careless or inattentive, and the consequences of that can be very serious. I propose that ALL government computers with Internet access be set up in such a way that NO downloads or clicking on links is allowed unless the communication is from a trusted sender, or the link is known to be a link to a trusted sender. This process should be entirely automated so that users simply cannot perform actions that are not allowed under these restrictions. That is, the list of trusted sites and links should be built into the system in such a way that restricted actions without explicit permission are not possible.

Permission for any restricted actions should almost always be automated. Any attempt to do a download or click on a link should immediately receive a yes (the attempted action is simply allowed to occur) or a no (the attempted action is blocked and a message briefly explaining the reason is sent to the user). We will call the software needed to implement the allow or deny actions permission software (PS for short). It is necessary to have PS for all users, since even a wary user might be fooled, for example, by a sufficiently well designed internationalized domain name (IDN) homograph attack. The possibility of such attacks has been known at least since 2002 [1]. Hackers

have devised increasingly more subtle versions of such attacks since then. Design of PS in such a way as to detect even the most clever IDN homograph attacks is an important unsolved problem.

PS is primarily aimed at preventing the download of malicious code from the Internet. There is a different security concern about downloads of classified information that could then be passed on to unauthorized readers. It is clearly insufficient to rely on security clearance vetting to prevent this, since failures in that process are hard to detect and can have very serious consequences. The great number of people that need to be vetted shows that failures in the process will be common. For example, Booz Allen Hamilton alone has more than 20,000 employees with security clearances. The infamous vetting failure with Edward Snowden shows how serious the consequences of even one lapse can be.

The goal is to guarantee that downloads of secret information be approved not only by the requester (whatever his security clearance status) but also in some other way. This means that any recording of secret data on hardware (thumb drive, DVD, CD, etc.) can never be done without some explicit approval from someone other than the originator, and can never be done without a log of what information was recorded. The natural way to do this is to disable the possibility of any download of secret data through a computer port unless explicit permission for this, and an accompanying log of what was downloaded, is obtained. It is a research problem to determine what changes in computer hardware would be needed to achieve this in such a way that the hardware could not be circumvented even by a sophisticated hacker. Standard security software (for example, see [2]) often provides ways for an administrator to block USB ports, but we believe software is insufficient and that hardware changes will be required for effective security (see Question 2 below).

Question 2 Cyberinfrastructure Needed to Address the Research Challenge(s) (maximum ~1200 words): Describe any limitations or absence of existing cyberinfrastructure, and/or specific technical advancements in cyberinfrastructure (e.g. advanced computing, data infrastructure, software infrastructure, applications, networking, cybersecurity), that must be addressed to accomplish the identified research challenge(s).

Question 2. First we deal with the issue of how to physically block USB ports from being used for nefarious downloads. Some naive solutions were proposed in the early days of USB (in 2003 someone [3] recommended putting glue in the port!). Dongles using hardware keys are not a solution because they depend on the honesty of the user. What is needed is a hardware solution that is relatively economical and not onerous for honest users. I propose the design of a new type of port, which could be called GSB (“G” for government). This GSB port would be on one end of a device, about the size of a thumb drive, whose other end would be a standard USB connection. The USB end would be plugged into the computer USB port which needed to be blocked. The connection would be an unbreakable one in the sense that the USB port into the computer would not function unless the GSB device was attached. It would in fact be difficult to remove the GSB device once it was installed.

The GSB device would only accept thumb drives or other external devices which were equipped with a unique identifier (much like an IP address) which had been authorized in advance to be connected to the chosen port. This authorization would only be valid for a certain period of time, as determined by the administrative authority for the computer in question.
Each external device authorized for connection to the GSB port would be known to have a tracking signal device (similar to the GPS devices used in cell phones)installed, and typically the external device would be associated with an authorized user of the computer. The external device would usually be stored in the same building where the computer typically was, and the tracking signal from the device there would be continuously monitored, so that any movement of the device would be known in real time. Allowable locations for the device would be registered and any departure into a location not allowed would generate an immediate alarm to the security people for the building. In particular, any removal of the external device from its usual building would need to be authorized in advance. If this authorization was not obtained, an alarm would be generated when the device left the building.

Next we consider the various problems which must be solved in order to be able to design and build the GSB device and external devices to attach to it, as described above. These problems include:

(a) Deciding how to securely attach the GSB device to the chosen computer USB port in such a way that removal of the GSB device without damaging the USB port would be very difficult, and would in any case generate an alarm.

(b) Designing the GSB device so that the built-in tracking signal device could not be disabled without destroying the GSB device.

(c) Setting up the protocols for the interaction of the GSB device with the administrator in charge of the device. This would include making changes in the ports and/or computers to which the device could be attached, making changes in the locations to which the device could be taken without generating an alarm, authorizing the device to be removed from the building, and responding to any alarms generated by the device.

REFERENCES

1. Evgeniy Gabrilovich and Alex Gontmakher, The Homograph Attack (http://www.cs.technion.ac.il/~gabr/papers/homogra ph_full.pdf), Communications of the ACM, 45(2):128, February 2002
2. https://support.symantec.com/en_US/article.TECH105770.html
3. http://www.techrepublic.com/article/disable-usb-ports-to-prevent-unauthorized-data-transfers/5030674/

Consent Statement

- "I hereby agree to give the National Science Foundation (NSF) the right to use this information for the purposes stated above and to display it on a publically available website, consistent with the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode)."